

AD-A071 697

IBM THOMAS J WATSON RESEARCH CENTER YORKTOWN HEIGHTS N Y

F/6 5/10

COGNITIVE PROCESSES IN DESIGN.(U)

APR 78 A MALHOTRA, J C THOMAS, J M CARROLL

N00014-72-C-0419

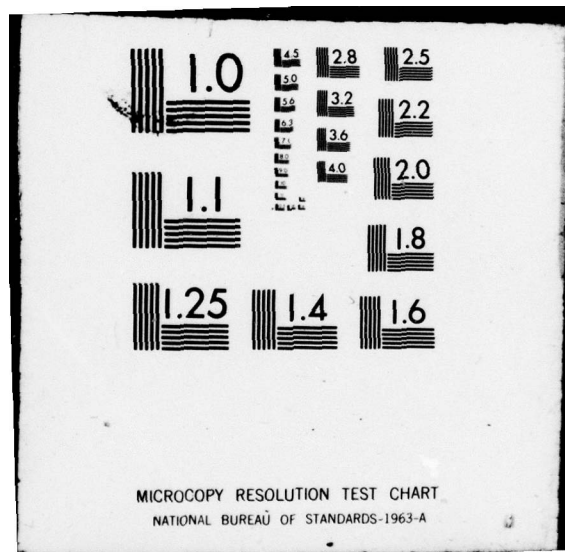
UNCLASSIFIED

RC-7082

NL

| OF |
AD
A071697





14 RC-7082 (#30392) 4/21/78 (Rec'd 6/1/78)
Computer Science 36 pages

9 **Research Report**

6 **COGNITIVE PROCESSES IN DESIGN**

10 Ashok/Malhotra, John C./Thomas, John M./Carroll and Lance A. Miller

I.B.M. Thomas J. Watson Research Center,
P. O. Box 218, Yorktown Heights, New York 10598

2
LEVEL 4

AD A 071 697

11 21 Apr 78

12 42p.

15 W44414-72-C-0419

DDC FILE COPY

IBM Research Division
San Jose · Yorktown · Zurich

349 250
DISTRIBUTION STATEMENT A

Approved for public release,
Distribution Unlimited

DDC

RECEIVED
JUL 25 1979
A


79 07 24 081

RC 7082 (#30392) 4/21/78 (Rec'd 6/1/78)
Computer Science 36 pages

COGNITIVE PROCESSES IN DESIGN

Ashok Malhotra, John C. Thomas, John M. Carroll, and Lance A. Miller

I.B.M. Thomas J. Watson Research Center,
P. O. Box 218, Yorktown Heights, New York 10598



ABSTRACT: This paper reports on a observational study and two experimental studies of the design process deriving from our interest in improving the design of computer software. A model of design is developed from these studies. This consists of three interacting processes: goal elaboration, design generation and design evaluation. The experimental studies show that design specifications are often incorrect and incomplete with different designers paying more or less attention to different aspects of the design. Finally, from the design model and the results of the experimental studies, a number of aids to the design process are developed and discussed.



This research was supported in part by the Engineering Psychology Programs, Office of Naval Research.

Reproduction in whole or in part is permitted for any purpose of the United States Government.

Approved for public release; distribution unlimited.



LIMITED DISTRIBUTION NOTICE

This report has been submitted for publication elsewhere and has been issued as a Research Report for early dissemination of its contents. As a courtesy to the intended publisher, it should not be widely distributed until after the date of outside publication.

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
DDC TAB	
Unannounced	
Justification	
By	
Distribution/	
Availability Codes	
Dist	Availand/or special
A	

Copies may be requested from:
IBM Thomas J. Watson Research Center
Post Office Box 218
Yorktown Heights, New York 10598

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RC 7082	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Creative Processes in Design <i>Cognitive</i>		5. TYPE OF REPORT & PERIOD COVERED Interim Technical Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Ashok Malhotra, John C. Thomas, John M. Carroll and Lance A. Miller		8. CONTRACT OR GRANT NUMBER(s) N00014-72-C-0419
9. PERFORMING ORGANIZATION NAME AND ADDRESS International Business Machines T.J.Watson Research Center, P.O.Box 218 Yorktown Heights, N. Y. 10598		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NR-197-020
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Code 455 Arlington, Virginia		12. REPORT DATE
		13. NUMBER OF PAGES 36
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This paper reports on a observational study and two experimental studies of the design process deriving from our interest in improving the design of computer software. A model of design is developed from these studies. This consists of three interacting processes: (cont.)		

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

20. (cont.)

goal elaboration, design generation and design evaluation. The experimental studies show that design specifications are often incorrect and incomplete with different designers paying more or less attention to different aspects of the design. Finally, from the design model and the results of the experimental studies, a number of aids to the design process are developed and discussed.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

1. INTRODUCTION

The Behavioral Sciences Group at the IBM T. J. Watson Research Center has been engaged in studying the cognitive activity associated with the design of complex structures and systems. Our effort has been directed at identifying and characterizing important design processes: the articulation of goals and requirements; the generation and evaluation of sub-solutions; and the integration of sub-solutions into a final solution. Interest in design drives from our motivation to improve the design of computer software. The work described here includes studies of other types of design situations in addition to software design. Nevertheless, all of the work is believed to be relevant to computer software design.

This paper describes an observational study and two empirical studies of the design process. These are contained in sections 2 and 3, respectively. A model of the design process developed from these studies is described in section 4. A set of aids to the design process, based on the design model and the results of the empirical studies, are discussed in section 5. Section 6 contains some recommendations for further work. This work provides a basis in cognitive theory for computer support of creative design.

1.1 Existing Theories of the Design Process

Theories of design can (with some notable exceptions; e.g., Freeman and Newell, 1971; Eastman, 1969) be categorized as follows: 1. Application specific models or suggested procedures (e.g., Walker, Gurd, and Drawneek, 1975), 2. General verbal models of the psychological processes involved in creativity (e.g., Prince, 1970), 3. General discussions of design followed by a formalism which purportedly allows automation of some part of the design process. (e.g., Alexander, 1964; Amkreutz, 1976). 4. A list of stages in design. (e.g., Jones, 1970; Metzger, 1972). While these approaches may be valuable to practitioners of design, they fall short in three respects from providing an adequate theory of design. 1. They

do not adequately differentiate design from other kinds of problem solving. 2. They do not provide a basis for empirical validation. 3. They do not provide a unified framework for viewing all aspects of the design process. A model of the design process that overcomes some of these limitations is developed in this paper. This model differs from most previous attempts to describe design as a sequence of steps in three ways: 1. It explicitly allows iteration and recursion. 2. It provides criteria for identifying phases of design. 3. It provides a framework within which specific hypotheses can be empirically tested.

1.2 Design Definitions

In order to facilitate the discussion of specific studies it will be useful to define some important terms. A *problem state* is said to exist when a human, or other goal-oriented system, has a *goal* but no immediate procedure that will guarantee attainment of the goal. The goal may be a satisfaction to be achieved or a dissatisfaction to be alleviated. *Problem-solving* occurs in moving from a problem state to a non-problem state. In problem-solving, then, a person begins in an initial state, uses transformations that move him from one state to another, and ends in a final state. Any of these states and transformations may be well-defined or ill-defined (see Reitman, 1965).

In the case of design problems, the person is generally not forced to start from a specific initial state, and although there may be constraints on what may be used, the transformations are usually not limited. In real-world design situations, the goals are, typically, fuzzy and poorly articulated and cannot be mapped directly into properties of the design. Thus, the exact configuration of the final state is not prescribed. A part of the design process consists of formalizing and refining the design goals into functional requirements that can be matched by properties of the design. Even so, it is usually difficult to tell how well a design meets a particular functional requirement. In addition, the functional requirements often cover different dimensions and the trade-offs between them are rarely well specified.

The properties of a design arise from a combination of the properties of the *design elements* that constitute it and the *design organization* that specifies their interaction. Design elements are available, indivisible units with given properties. Designs consist of some number of design elements interacting with each other in a *design organization*.

2. CLIENT-DESIGNER DIALOGS: AN OBSERVATIONAL STUDY OF DESIGN

This section reports on a preliminary study that consisted of videotaping actual problem-oriented dialogs between real people, playing real roles and attempting to solve real-world problems. The analyses are based on transcriptions of the videotapes.

The scenario for this class of design processes is that a client (C) confronts a consultant, expert or designer (D) with a problem and the expert is expected to provide a solution to this problem. In the cases we are interested in, the solution takes the form of a complex system or structure.

In this section we describe a preliminary study of client-designer dialogs. In a typical dialog, a client with a manifest problem meets a designer with skills in a particular area to discuss a means of solving the design problem.

If we idealize the client and the designer, then the former can be considered to have only design goals and the latter only solution components. In the abstract then, a Client-Designer (C-D) dialog can be considered to consist of the translation of design goals into a set of functional requirements that the design must meet and the generation of a design to meet those requirements.

Actual C-D dialogs are, however, considerably more complex. They include the examination of (partial) proposed designs to test if they violate some unstated goal, the domination of a design solution by a better one and the consolidation of design components into a final design.

2.1 Dialog States

We found that the C-D dialogs consisted of a series of "cycles"; each cycle consisting of a regular succession of "states". A state is defined as a portion of the dialog that is oriented towards a single purpose. The states are presented below, along with the main speaker(s) in each state

STATE	MAIN SPEAKER
1. Goal Statement	C
2. Goal Elaboration	C&D
3. (Sub) Solution Outline	D
4. (Sub) Solution Elaboration	D
5. (Sub) Solution Explication	C&D
6. Agreement on (Sub) Solution	C

Each of these states may consist of one or more uninterrupted speeches by C or D. States cannot coexist in time. State 1 consists of C stating one or more design goals. In state 2 these goals are developed either by C himself or, more usually, in response to questions from D. This phase may consist of exploring goals in more detail, establishing the context within which they exist or discussing contiguous requirements. Since they deal with the same material at different levels of detail, states 1 and 2 are often difficult to differentiate. Typically, however, C provides rather general, subjective goals in state 1. In state 2, D pushes towards greater clarity and objectivity of these goals.

State 3, which is usually brief, consists of D (and sometimes C) suggesting a sub-solution to the problem. The solution is usually suggested in outline at an abstract level. The following fragment from a dialog aimed at designing a better address book illustrates a rapid succession of states 1, 2 and 3. In this and subsequent quotations, equipment and system identifiers have been changed.

STATE 1

C: Right. Right. But the real question is, when you carry a book around, it should be compact. You can't carry a printout.

D: MmmMmm

STATE 2

C: And, if you have it formatted, you use only the left portion of the page, so I much prefer to carry around this sort of thing rather than the full.....

STATE 3

D: Have you done any experiments yet with the XXXX printer to try printing it with a...perhaps a telephone book type fonts or something of this sort, where you could pack more information into a small page?

State 4 consists of the designer developing the details of the suggested sub-solution and explaining its properties. In state 5, C joins him and together they explore the properties of the suggested solution. Since states 4 and 5 deal with the same material they may also be difficult to differentiate. However, state 4 deals primarily with the properties of the design per se while state 5 includes consideration of the interaction of the proposed design with the context of the client.

We like to think of a suggested solution as a physical object with a large variety of interesting surface features. With this analogy state 5 can be compared to rotating the object in different directions to familiarize C with all its features and properties and ensure that he realizes and appreciates their implications for his particular situation. State 6, which may or may not follow states 4 and 5, consists of the client agreeing to a particular sub-solution and affirming that it meets the design goals and functional requirements. The following dialog fragment illustrates states 3,4,5, and 6:

STATE 3

D: O.K. That's a good idea. We may, we have NXNN control units downstairs. Its quite possible that we could have a line removed from the XNNN control unit, upstairs (hum) installed on the NXNN on the AA system downstairs. If we can do that it might be possible to access IIII through the AA machine -- which would make it virtually accessible from anyone in the whole building.

STATE 4

C: That may be simply to protect the password. Having to go through the my userlib.

D: Right and being on AA, if its running on a virtual machine under AA, all of your ... all or any of your output can be spooled and printed at the local printer.

C: Ok. That ... that interests me more than doing anything with the printer. If that ... if that's feasible I would like to leave the printer where it is.

D: Uh-uh.

STATE 5

C: Fine. Leave it alone. That's fine, if the guy's in the library and he wants a little thing and he wants it here and now ...

D: Right. Now. These scopes go through the building network to the scope control units downstairs into AA.

C: That says the modems can be done away with too?

D: Yeah. Then everything goes.

C: You wouldn't need either of those?

D: Right. You wouldn't need either your control unit or your modem.

STATE 6

C: Ok. Can we do something on it? See what we can do? Forget about the printer. Forget about the rest of this jazz and just move in the direction of having ...

D: Right. Sure.

C: ... direct output.

If state 6 does not follow states 4 and 5, i.e. if the client does not agree to a suggested solution, it is because he finds that it violates a stated or unstated design goal. In this case state 5, solution explication, is succeeded by state 2, goal elaboration, with the client elaborating the violated goal.

2.2 The Structure of the Design Cycles

Examination of the dialogs reveals them to consist of a succession of design cycles. The basic cycle consists of a regular succession of states 1 through 5, possibly followed by state 6. There are few exceptions to this structure if we allow for degenerate design cycles in which states 4 & 5, solution elaboration and explication, are skipped (in case the solution is easily understood and obviously appropriate) or states 1 & 2, goal statement and elaboration, are skipped (in case a sub-solution is rejected and another solution proposed without any discus-

sion of the violated requirement). Figure 1. shows the cyclical structure of a design dialog.

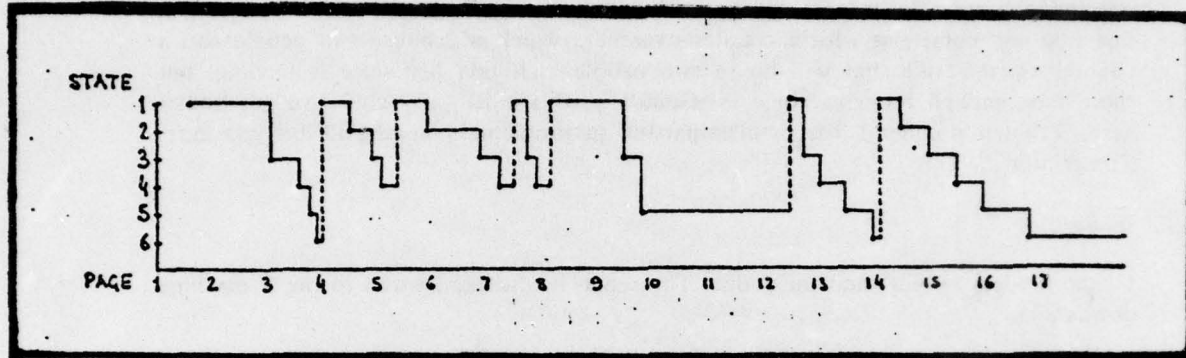


Figure 1. The Cyclical Structure of a C-D Dialog.

"The Day of the Jackal", a fictional work by F. Forsyth (1971), contains a conversation between the protagonist who needs a gun with special features and an expert gunsmith (pp 67-72). This conversation is a C-D dialog as defined here and it displays the same cyclic structure exhibited by the experimental dialogs. The client starts by describing his requirements and a design is generated in five cycles.

2.3 The Contents of the Design Cycles

Further examination of the dialogs reveals, however, that beneath the regularity of structure lies a diversity of content. At the simplest level, a cycle may start with a fresh set of requirements (at the start of a dialog or after an acceptable sub-solution) or it may start with merely an elaboration of previously mentioned requirements (after a rejected sub-solution). Similarly, there are differences in the extent and character of solution elaboration and explication depending on specific circumstances. At a deeper level, although solution suggestions and discussions always follow discussion of requirements, the solution that is outlined and discussed need not cover the requirements that precede it.

Consider, for example, a part of the first cycle of the last dialog quoted above. The parenthesized words are uttered by the actor who is supposed to be listening.

C: Now its a crummy arrangement for four people. There are a number of problems involved in this. One is, what are we going to do to physically house these four people. I've got to get some one who knows how to draw a little bit better than this to come out and give me a design that will move the card catalog (All right) and the people on this side and come out with a counter over here which will house two people and a counter on this side that will house two people. (Right) My scale is terrible, but there'll be enough room to come in. Generally, this is it. all right? I've got bodies here. Problem number 1, the simplest part of the problem, is moving all this jazz here. The printer ...

D: Right.

C: the modem (Right) and the scope. The scope is also hard-wired to the T machine downstairs.

D: OK. We don't have any problem with moving scopes around on the turret (Now) because of cable connections. (Now control unit)

Four requirements are discussed in this cycle: limited space, moving the printer, moving the modem, and moving the scope. The solution proposed is that there is no problem with moving scopes around. This covers only one of the requirements. Presumably, the other requirements are stored in the designer's memory and covered by sub-solutions proposed to subsequent design cycles. In fact, in this dialog, solutions proposed in subsequent design cycles cover the other three requirements in the first cycle as well. In other design cycles in the dialogs, the solution proposed in state 3 matches the requirements discussed in the corresponding states 1 and 2 very well.

2.4 Discussion

Due to the articulation of at least a part of the design process, the C-D dialogs tell us something of the the individual processes that participate in the creation of a design. Broadly speaking, these correspond to the states defined above: the requirements are stated, developed, examined, and understood (states 1 & 2), partial solutions are outlined (state 3), developed and carefully examined and possibly accepted (states 4, 5 & 6). New requirements are often uncovered in the process of examining solutions and these may start another design cycle. Some kinds of requirements, such as human factors, are hard to formalize and articulate. These can be discussed indirectly through a study of proposed solutions.

The elaboration and development of the client's goals leads to a set of functional requirements that must be met by the proposed design solutions. Generation of design solutions in the C-D dialogs seems to consist of attempting to find design elements to meet functional requirements and then tying these together into a coherent design. This corresponds roughly to what is called "bottom-up design". While this is not the only design strategy exhibited in the dialogs, it is the predominant one and it seems to be encouraged by the fragmentary presentation and discussion of requirements. In some real situations, particularly where complex designs are required, the designer would not have to generate a design on the spot. He would take the requirements away with him, have an opportunity to internalize them and use different kinds of strategies to generate a design.

The underlying bases for the cyclic structure of the C-D dialogs and the lack of coordination between the requirements discussed and the solutions proposed in a cycle are not immediately obvious. Why does the designer propose partial solutions when he knows he does not have complete information on the problem? Why does he not wait until the client has enumerated all his requirements? There may be a number of possible motivations for this behavior. When two people talk, a dialog seems to be expected and the designer has to contribute to it. Since his domain is solutions he proposes solutions. Also, he wants to create a certain amount of confidence in the client. This may best be done by suggesting and discussing solutions. But the important reasons for the premature introduction of solutions seem to have to do with the designer helping the client articulate and elaborate his goals. In the following fragment of a dialog the designer suggests a solution and in the ensuing dialog is able to learn more about the client's requirements.

D: Right. I know that's a problem, the clanking, with this type of printer, it's a noisy device.

C: It doesn't clank, it buzzes.

D: Its a noisy device. In a limited area, like the one we're talking about right now, its hard to work with. (Yep) I understand that. There's another type of printer, there's a NNNX printer which is ... It would give you every line of input and output simulta-

neously -- Instead of you asking for a particular line to be printed, you would get everything and then if you asked ...

C: I don't follow you.

D: O.K. Every entry that you make on the scope (humh) will be printed on the NNNX. Every answer that you receive on the scope, or every screenful of information that you get (humh) for an inquiry, will print on that NNNX. So it is in effect a hard copy device.

C: Sentence printer.

D: Right. Now, that might be easier to go to but like I said before if we wanted to take your original suggestion, (that's) I would have to consult with software.

C: That's, that's gonna that other way's gonna spit paper out like there's no tomorrow. Cause then theoretically when the guy comes on he'd have to have the printer on in case he wanted something ...

D: Uh, uh.

C: ... and then he'd have to wade through the paper and say "Ah, this is the one I want" and tear that one out.

D: Right. The printer would be on at all times.

C: O.K., No, Now that's not what I'm after. I'm after the selective way where the where the guy says "O.K. I want this ..."

D: Hmm-hum

C: And we can take the guy, he can have 100 documents, 200 documents, but have them today as opposed to waiting a week for them.

In the somewhat more complex case quoted below the designer is able to suggest an alternate means of meeting some of the client's goals. In other words, the designer is able to suggest different functional requirements for the design.

D: Well, let's back up a bit. First, you're doing a sort here...

C: Yes

D: and yes, I admit that the machine is very capable of doing that kind of sort....it costs nothing, just about....but to a certain extent the difficulty is not so much that the machine has any cost involved in doing the sort, but rather that somebody has to remember how to do the sort, and you are thinking, I guess, not of just yourself and your secretary who could learn a procedure and that would be one person, but if half a hundred or a few hundred people are doing it....

D: . . . they are going to have trouble, so why not eliminate the sort procedure entirely, and just say that look, this is a file like other files that you've worked on, just like a letter or a document, and it has a first page and a second page and you enter the entries in a sorted way; you may change it that way, and if you have to change an entry for a guy whose name begins with N ...

In another instance the designer suggests a software solution to some requirements that the client assumed would be solved by hardware. This surprises the client. Somewhat later, discussing other requirements, the client suggests a software solution to encompass both sets of requirements.

2.5 Conclusions

We found that the C-D dialogs are structured into cycles each of which consists of introducing some requirements, discussing them, outlining a solution and elaborating and exploring it. The solutions proposed in a cycle do not, however, always match the requirements discussed in it. Unsatisfied requirements are carried implicitly into subsequent cycles. This observed behavior is used to construct a model of the design process in section 4.

3. TWO EXPERIMENTAL STUDIES OF DESIGN

3.1 The Restaurant Design Study

In the restaurant design study, in contrast to the C-D dialogs, no data was recorded during the design process. Rather, the designs produced by the subjects were measured in various ways. In this experiment we were also interested in attempting to develop some objective measures of the originality and practicality of the design --- the two factors generally conceded to constitute creativity (e.g., Stein, 1974). We were also interested in studying the characteristics of the design of *structures* in contrast to the design of *procedures*, as in software design (see section 3.2).

3.1.1 The Method

College students from a local liberal arts college served as paid subjects. Each subject was given a wealth of background data concerning a particular Bonanza restaurant that was

proposed to be located in an old church. Information given to the subjects included financial background data, site information, a floor plan of the church, and various financial and spatial requirements. Each subject worked alone to produce a design for the restaurant which they were allowed to communicate to the experimenters in any manner they wished. After the completion of the design, each subject was asked to fill out a questionnaire which asked about the goals of the design, the strategies used, the importance of various information sources, and for background data about the subject including SAT scores. The materials for half the subjects included a list of words. These words had been found in a previous experiment (Thomas, Lyon, & Miller, 1977) to aid in the solution of two simpler design problems. This word list was simply included in the materials; subjects were given no specific instructions with regard to its use.

A measure of originality, O , was defined according to the following procedure. First, the design of each subject was scored for the presence or absence of every feature present in any subject's design. Then, looking across subjects, the probability of each feature appearing in a design was calculated. In this manner, $H(i)$, the amount of information present in each design was calculated in the usual manner. Then, the maximum amount of information $H(\max)$ that could have been present in a design, given the data, was calculated. Finally, each subject's score, O , was defined as the ratio $H(i)/H(\max)$.

The other measure of performance was practicality, P . This measure was computed according to the following procedure. The goals of a restaurant design were decided upon. The top level goals were: satisfaction to the customer, satisfaction to the owner, and satisfaction to the employees. Each of these was further differentiated until a level of functional requirements was reached. At this level it was fairly easy to determine whether or not a particular design met a particular goal. Subjects' designs were matched against this goal tree and the design was credited one point for every node in the goal tree that it satisfied. (Higher level nodes were ANDs or ORs of the branches below it). Each subject's score P , was the fraction of the nodes in the goal tree that were satisfied.

3.1.2 The Results

We were interested in the trade-off between *O* and *P*. Other things being equal, it was felt that a subject would tend to increase the originality of the design at the expense of practicality or vice versa. However, a subject of higher ability might be able to score higher than one of lower ability on both *O* and *P*. Subjects were, therefore, split into six groups on the basis of their SAT scores. (One group consisted of subjects who did not report their scores). Within each group, there was a negative correlation between *O* and *P*. A two-tailed sign test makes this result statistically significant, ($p=1/32$). The distributions of *O* and *P* are also interesting (see Thomas, Malhotra and Carroll, 1977). *P* varied from .34 to .71 and was fairly normally distributed. The distribution of *O* was skewed and only varied from .61 to .77 with the majority of the subjects only varying from .64 to .70. Thus, only a few subjects produced highly original designs.

Also of interest were the subject's answers on the questionnaire concerning their strategy. Surprisingly, a majority of the subjects claimed to have designed top-down, to have planned their approach, and to have tackled the more difficult problems first. However, none of these self-reported strategy variables were correlated with either *O*, or with *P*.

What was predictive of *O* and *P* were the subject's expressed goals. Some subjects claimed to be more interested in having a design that was novel, imaginative, and original. These subjects scored significantly lower on *P*. Other subjects strived for a design that was workable and practical; they scored significantly lower on *O*.

Subjects whose booklets included the word-list aid scored significantly higher on *P*, but not on *O*. Perhaps, part of the explanation for this difference is that subjects who had the word list included in their materials were significantly more likely to include a verbal explanation. Thomas, Lyon, and Miller (1977) also found improved performance for subjects using this aid. The effectiveness of the aid is based on the hypothesis that, in design, subjects often possess relevant knowledge which is not spontaneously accessed. Much of this information will, nevertheless, be recognized to be relevant after its cued recall.

3.1.3 Discussion.

The results of the study suggest that in this problem most subjects at least claimed to have used 'good' strategies spontaneously. The exception is that some subjects seemed to have been so concerned about being original that the practicality of their design suffered (without an attendant increase in originality). Other subjects were so concerned with practicality that the originality of their design suffered (without a corresponding increase in practicality).

From a methodological point of view, the objective measures of *O* and *P* seem promising. Since these measures were systematically related to each other as well as to the subjects' expressed goals, further development of these measures should involve configurational information.

3.2 *A Study of Software Design.*

Four subjects were asked to write a set of functional requirements for a query system to retrieve information from three related personnel files. The organizational setting and the environment within which the system would operate was described as well as the background of the people who would be using the system and the kinds of problems they would need to address. Some sample queries were also provided.

The subjects, each of whom spent an average of three hours on the problem, were expected to specify the functions provided by the query system as well as the syntax for specifying the queries. Two of the subjects did this. Their approaches were, however, at the opposite ends of a spectrum. One of them provided a minimal set of functions with a very simple syntax. The other provided a sophisticated function set with a correspondingly complex syntax.

The other two subjects took completely different approaches. One said that a standard query language such as Query By Example (Zloof, 1977) could be used. The other provided a menu of queries for the user to select from.

In light of the model of the design process discussed earlier, the subjects were provided with top-level goals and came up with widely divergent sets of functional requirements. Thus,

the sub-goals or solution strategies generated for the higher-level goals seemed to vary widely and there did not seem to be an orderly procedure for generating the strategies. In fact, the sub-goals suggested by people seem idiosyncratic and to depend strongly on past experience.

3.2.1 Analysis of Software Designs

In a follow-up to the above experiment, eight subjects were provided with specific functional requirements for a query system and asked to design the data structures and the algorithms to meet the specifications. They were asked to specify the details of how the program would operate in a manner that a programmer could program from.

Comparative analysis of the designs reveals a number of interesting features. To begin with, the designs were all very different from each other along several dimensions. In overall measures, the subjects took between 3 and 6 hours to complete the designs. The number of sentences in the designs varied from 95 to 199 and the number of words from 706 to 2345.

Consider now the content of the designs. Since the functional requirements specified the syntax of the input to the program as well as the stages in which the user would specify the query, the top-level structure of the program was fairly uniform. In fact, all subjects organized the program into modules in one of two closely related ways. The detailed contents of the individual modules was, however, very different and covered a wide range of data structures and algorithms. Consider, for example, the data structure used to store the criteria to select the records to be displayed. Two of the subjects did not specify this data structure at all. Of the six who did, each of them used a different structure: a vector, a PL/I data structure, an attribute-value structure, two kinds of tables and a list with subsidiary switches. Each type of data structure had to be initialized and routines had to be provided for storing information in it and using this information for the data selection algorithm. What is even more interesting, when the subjects were interviewed subsequently they did not have strong reasons for selecting the structures they described.

The diversity in algorithms was equally striking. Consider, for example, the algorithm used to select the information to be displayed from the data files. If the various design

possibilities that are available at each stage are expanded into a tree we obtain a tree with 21 leaves. In other words, there are 21 correct forms that the algorithm can take. Each of these forms represent a series of design decisions such as the order in which the files are read, the manner in which selected information is noted, etc. and can be rated on a measure of efficiency. Of the eight subjects, one did not discuss the algorithm at all. Another merely stated its purpose but did not specify the details of its operation. Of the remaining six subjects, two specified essentially the same algorithm operating in a very simple but inefficient manner. The remaining four algorithms were all more complex and efficient but spanned the spectrum of possibilities. All of them were quite different in philosophy.

In addition to being very different, the designs contained errors, inconsistencies and unwarranted assumptions. Examples:

In one of the designs there would be no output if a selection criterion was not specified, i.e. if a user wanted all records of a certain type.

In another, the output would be in the sequence the records were stored in rather than the desired sequence.

One of the designs required a blank to be typed after the last output field.

A particular vector was variously referred to as "7 place" and "10 place".

The level of detail in the specifications also varied considerably. This variation was as marked between modules in the same design as well as between different designs. As an example of the variation observed, one subject stated the algorithm to retrieve the data to be displayed as "RETRIEVE checks each record in the data against the query...and adds to DLIST..." while the other subjects specified complex retrieval algorithms in considerable detail over many pages. Clearly, the former specification is not detailed enough to program from.

What are we to make of this great diversity in the designs? Compare the situation to engineering design. Presented with eight designs for a building or a bridge one could define

appropriate performance measures, such as construction time and cost, test whether the designs met functional requirements such as accommodation and maximum load and rank the designs on these multidimensional attributes in some manner. Some of the measures, such as aesthetics, would be subjective but, given an appropriate representation, each judge could quickly assign scores for them.

For software designs, performance measures can be defined without much trouble but, because the designs are not executable and cannot be tested, it is very difficult to assign scores for these measures. It is very difficult, in fact, to ascertain whether the design is complete, is consistent, or whether it meets the functional requirements. This seems equivalent to not being able to tell if a building will stand up!

Another contrast between engineering and software design is the level at which the designer works. The engineer designs in terms of rooms, cantilevers, trusses and other structural elements. The software designer builds data structures and algorithms from the basic facilities of the programming language and the access methods. Programming languages and access methods have been designed to be as general and powerful as possible. One can do literally anything with them and they do little to constrain, suggest or facilitate good, appropriate or preferred data structures and algorithms. Thus, the software designer can create a bewildering array of potential pieces from which the program can be put together.

This diversity of potential intermediate components and the difficulty of evaluation makes software design difficult and the designer often picks intermediate structures on arbitrary bases. Familiarity is probably the most important of these. Furthermore, comparing competing data structures or algorithms is difficult and it is usually possible to find several examples that all perform adequately.

There has been some work done in developing, formalizing, classifying and making available good data structures and algorithms for various purposes (see, for example, Knuth, 1969a, 1969b) but each data structure and algorithm usually has to be programmed from scratch for each program that uses it. The state of the art today is such that it is only in rare

cases that a commercial programmer uses a standard subroutine to do a part of his processing. Although there have been some proposals for facilities that allow programs to be built from standard reusable modules (see, for example, Loewner, 1969; Morrison, 1971) there is no equivalent to the off-the-shelf assembly in the software world.

3.2.2 Design Styles

The design specifications seem, fortuitously, to have two quite different styles, with an equal number of subjects using each style. The first, the "programming" style (PS), is characterized by a highly formatted text (indentations, multiple line-skipping to create separate segments, etc.) and use of programming transfer-of-control constructions (use of labels, "goto" statements, "if-then" statements, etc.). The writing is, typically, terse, objective, and impersonal.

The second style, the "narrative" style (NS), has none of these features. Instead, it is written much like normal continuous text material. The writing is discursive, subjective, and often personal (through the use of pronouns).

We were interested in determining whether this qualitative judgement of styles would be supported by more quantitative measures. To determine this we examined objective measures of word usage. These are shown in Table 1. Numbers, other than the integers 1 to 10, were not counted as words. The differences in total words, total unique words and words per

sentence indicate 50 to 60 percent greater verbiage and lexical size for the NS group.

STYLE	PROGRAM (PS)	NARRATIVE (NS)
Average Time Spent	4.94 hrs	3.81 hrs
Total Sentences	468	584
Total Words	4216	6778
Total Unique Words	610	1022
Average Words Per Sentence	7.14	14.07

Table 1. Differences Between The Two Styles

An examination of the 150 most frequent words -- accounting for 70-77 percent of the total words used -- revealed that only about half of these words were used in common. At least one-quarter of the non-common words were not used by the other group at all. This indicates strong differences in the choice of the words used by the two groups to describe their designs. It also implies extensive differences in syntactic constructions due to the lexical constraints that are associated with co-occurrence of words in sentences.

Further, observations of the apparent part-of-speech classes of the 150 most frequent words suggested more evidence of stylistic differences: PS appeared to use fewer adverbs than NS; NS used a much wider variety of tensed-verb forms (e.g., present participle, past participle, simple past, more frequent use of modals, etc.); and NS used roughly three times as many verbal quantifiers (e.g., each, all, some), deictics (e.g., this, those) and personal pronouns. All

of these differences support the hypothesis that there were two measurably distinct styles in our data.

Some differences in these indicated measures of style might be expected even with a random division of the subjects into two groups. Nevertheless, these differences in word usage --- and the implied syntactic differences --- were not found in a previous study of natural language procedures. In this study, while there were differences in the algorithm specified there appeared to be only a single linguistic style (Miller and Becker, 1974).

Quantitative analysis of the data obtained in the software design experiment is continuing. Pending this analysis it is difficult to make assessments of relative design quality for the two styles. It did appear, however, that the designs of the PS group were somewhat better organized and more detailed than those of the NS group.

4. A PROPOSED MODEL OF THE DESIGN PROCESS

A model of the design process emerges from the client-designer dialogs discussed in section 2 and from our observations of the experimental studies discussed in section 3 in which the subjects played the roles of both client and designer in the intermediate stages of design. This model attempts to decompose the design process into more fundamental underlying processes. Three such processes are postulated, each with its initial and final states. The first process may be called *goal elaboration* and consists of the statement and discussion of the goals of the design. This corresponds to states 1 and 2 of the C-D dialogs. Typically, C begins with some fuzzy sense of there being a problem. This is the initial state for this process. Usually, C explores the problem for some time before he talks to D. During this gestation period he attempts to examine his underlying motivations and decompose them into potential sub-solutions or sub-goals such that satisfaction along one or more of these components would lead to satisfaction along the parent goal. Typically, a number of these goal-components can lead to satisfaction along the parent goal. C selects from among them on the basis of effectiveness, efficiency, elegance or whatever criteria he may deem appropriate.

There are many ways to think of this decomposition of a goal into components and selection from among the developed components. The components may be thought of as sub-goals as we do or as solution approaches or strategies. Duncker (1945) calls these the "by-means-of-which or functional value". To meet the parent goal, the selected solution strategies have to be implemented and this gives rise to the sub-goals.

To return to the model, C continues the process of goal decomposition and sub-goal selection until the sub-goals are specific enough to be considered as functional requirements that can be matched by properties of the design. This is the final state of the goal elaboration process.

In fact, however, C rarely has the necessary specialized knowledge of the design discipline to carry goal decomposition down to the level of functional requirements. Clearly, it is difficult to come up with solution approaches unless one knows what kinds of solutions are available. At some point, therefore, D or someone with detailed knowledge of the discipline within which the design is going to be realized has to help the client elaborate his goals to the level of functional requirements. This corresponds to state 2 of the C-D dialogs.

The functional requirements represent the point at which the responsibility for the design process shifts formally from C to D. In some cases a contract may be drawn up stating the requirements that the design must meet. The formality is more legal than intellectual, however, and in practice it is very difficult to pin down exactly when the goal elaboration process matures into a set of functional requirements. Thus, in fact, the functional requirements may be drawn up at any stage of the goal elaboration process after C and D start to work together.

The second process, *design generation*, starts with the functional requirements and attempts to come up with a design organization and design elements working together within the organization such that their combined properties meet the functional requirements. This corresponds to states 3, 4 and 5 of the C-D dialogs. If multiple designs are produced then the one that best meets the functional requirements is selected. A number of different strategies

may be used to cope with the complexities of the design environment and to actually produce the design. There is evidence (e.g., Walston and Felix, 1977) that the quality of the design depends strongly on the strategies employed. In general, performance differences in complex problem solving tasks are correlated with strategy differences (see Meister, 1976).

When a (partial) design is generated, the third process, that of *design evaluation*, begins. This corresponds to states 5 and 6 of the C-D dialogs and consists of discussing the various properties of the design and how well it meets stated or unstated goals. This process operates within the design generation process. It starts with the introduction of a new (partial) design and ends with either its acceptance or rejection. The most important feature of this process is that it may uncover new requirements, especially those that are fuzzy and hard to formalize.

In a study of the importance of representations to design generation, Carroll, Thomas and Malhotra (1978) found that subjects who had access to a suitable representation were able to generate better designs faster. We hypothesize that a good representation reduces complexity by representing only the important features of the design. Also, it provides a framework within which the designer can integrate sub-solutions and test proposed designs. A poor representation may, however, lead to a deterioration in design performance by ignoring essential variables.

The goal elaboration process and the design generation process coexist within the design process. They are usually inextricably interlaced and they assist and strengthen each other; one fading into the other only to rise again from it a little later. Design evaluation occurs, as required, within the design generation process. Information may be obtained in any one of the processes that could restart any of the other processes and be used by them.

The data obtained from the C-D dialogs and the experimental studies supports the existence of these three processes and their postulated relationships to each other. In the following sections we examine the implications of this model and use them and the results of the experimental studies to define potential aids to the design process.

5. AIDS FOR THE DESIGN PROCESS

Our motivation for studying design was to understand the kinds of aids that can assist the design process and make it more efficient and more effective.

The results of the experimental studies and the design model described in section 4 suggest the characteristics of aids to the various sub-processes within the design process. The following three sub-sections describe aids that assist the basic sub-processes of goal elaboration, design generation and design evaluation.

5.1 Aids to the Goal Elaboration Process

A Delphi survey of programming managers indicated that these managers felt that communication problems between clients and designers were the biggest source of problems in software (Scott and Simmons, 1974). An extensive correlational study (Walston and Felix, 1977) indicated that the complexity of the interface with the customer, degree of client participation in the definition of the requirements, and the percentage of programmers doing development who participated in design of functional requirements constituted three of the five most predictive variables of programming productivity. The complexity of the customer interface was three times as important as the use of structured programming.

Such studies indicate that a better method of communicating goals from the ultimate users or buyers of programs and the software designer could *very significantly* impact the cost of software and hence total system cost.

One strategy for assisting the goal elaboration process is suggested by the C-D dialogs. In these, the designer describes a number of designs, that may or may not be appropriate, to the client. This seems to help the goal elaboration process by showing the client what is available and allowing him to select certain kinds of designs for further investigation. Such an aid is not difficult to design in certain domains. The prospective owner of a restaurant can, for example, be shown slides of aspects of different kinds of restaurants. This will give him some feeling for the design space and he can then concentrate on exploring in depth the few alternatives that interest him. Similarly, a well-stocked information facility could present different kinds of software designs to meet a stated need. The difficulty in this case is to

classify the designs by purpose and to select appropriate ones from a library based on some statement of the design goals.

The goal elaboration process requires the client to access his memory for sub-goals or solution strategies. This is often difficult as information can be conveniently accessed only according to how it is stored and it is difficult to anticipate all the contexts that a given piece of information will be useful in at the time that it has to be stored. Retrieval can often be assisted by unstructured memory cues that bring different kinds of information into focus. Some techniques for providing such cues are discussed in the following sub-section.

5.2 Aids to Design Generation

Aids to the design generation process should assist the designer in generating design elements and design organizations to meet specified functional requirements. In many cases, suitable design elements and organizations may exist in the designer's memory and the purpose of the aid should be to help him retrieve the information more efficiently and effectively. In other cases, the designer may have to search for design elements and organizations in the outside world and a quite different kind of aid is required to support this process.

5.2.1 Unstructured Aids

In the software design experiment we found that subjects picked the structure of individual modules on an arbitrary basis --- often directed by past experience. In the process of design generation, the designer takes the functional requirements and attempts to find design organizations and design elements to meet them. His success at doing this depends on the information stored in his memory, the organization of that information, and the algorithms that are used to access it. In other words, portions of the design are generated from the contact between information stored in the designer's memory and the stimulus of the functional requirements.

There is some evidence that the use of an unstructured, domain-independent aid such as a list of random words increases the effectiveness with which memory is searched. Thomas and

Lyon (1978) conducted an experiment in which subjects were asked to name the items of information that they expected to appear on an invoice. Subjects were rewarded on the number of correct items they could name. Thomas and Lyon found that subjects did considerably better if they were mentally cued in certain ways. For example, they were asked to imagine the uses that the people who worked with the invoice would put them to, they were asked to come up with items starting with each letter of the alphabet, and they were asked to read a random list of words. The number of items named by the subjects increased after each set of cues. Their performance after using the aid was much better than their performance without it. In the restaurant design experiment the subjects who had random word lists included in their folders seemed to score significantly higher on practicality than the control group. The hypothesis is that the unstructured aids tap into different parts of memory and so increase the amount of memory that comes into contact with the requirements.

The advantage of unstructured aids is that they are simple to implement and that they are domain independent. It should not be very difficult, therefore, to devise a set of simple unstructured aids, modelled, perhaps, after ELIZA (Weizenbaum, 1969), that would provide cues to the designer. If he were to use the aids, evidence indicates that the quality of his designs would be greatly enhanced.

5.2.2 Collaboration

In the restaurant design experiment, individual subjects fulfilled between thirty and seventy percent of the functional requirements. Each of the requirements, however, was fulfilled by some subject. This seems to indicate that some subjects spent more time, had more experience, gave greater emphasis, or were better at meeting one kind of functional requirements than another. A pair of these subjects, chosen for complementary skills, would, presumably, be expected to create a better design than either subject by himself. A C-D dialog can, in some sense, be viewed as a collaboration between two individuals with complementary skills and it leads, presumably, to a better design than either participant could create on his own.

Thus, in general, if the requirements of a design can be decomposed into a few general categories requiring specific kinds of expertise, and if a group of individuals can be selected to have special talent or experience in these areas then this group would be expected to do better than a single individual. The use of interdisciplinary teams for difficult situations is an example of this method (see Papanek, 1974).

5.2.4 Complementary Aids

In another direction, if the complex of skills required for a particular kind of design could be identified then the individual designer could be aided or tutored in the skills that he was deficient in. For example, if it were possible to predict, on the basis of standard tests of component skills, those subjects who would score high on originality or practicality in the restaurant design experiment then these subjects could be aided on complementary skills. Subjects who were expected to score low on practicality could be tutored on cost estimating, accounting and the operation of a restaurant; subjects expected to score low on originality could be shown examples of existing restaurant designs and assisted with unstructured aids.

5.2.5 Structuring Aids

The negative correlation between originality and practicality in the restaurant experiment may be explained by the time subjects spent making their designs original or practical. Thus, designs may be incomplete because the designer does not consider all aspects of the problem.

In the C-D dialogs the client seems to lead the designer through the design process. A structuring aid that would similarly lead the designer through the design process could contribute to the completeness of the design by directing designers to work on all aspects of the problem. Such an aid may ask the designer to generate his design in prescribed phases or merely to specify it in a particular format. In the former case, the aid will impose a design strategy on the designer and he may resent this. Also, unless the aid is very sophisticated, its repertoire will be limited to a few design strategies. It will select one of these in each situation, based on a superficial analysis of a few variables. Since the selection of a good strategy in a

design situation is a complex and little understood function of the characteristics of the task and the designer the aid will often prescribe a strategy that is poor for the situation. The user, forced by the aid to use this strategy, may do worse than he would have done on his own.

5.2.6 An Information Facility

To assist the designer in searching for suitable design elements and organizations an information facility can be developed which could present detailed organizations of designs, the design elements that participate in those organizations, how the elements interact with each other, and the desirable ranges of their properties. What we are proposing is, in some sense, an interactive version of a design handbook.

A handbook, typically, provides standard designs and the parts required for those designs. It may also provide a "rationale" for each design as well as its advantages and limitations and the situations in which it is appropriate. The facilities of the computer would allow a much larger number of designs to be stored, a much more flexible and powerful system of classification, automatic rules for searching the information as well as much more detail on each design than would be possible otherwise.

Such an information facility would also contain the various types of design elements that are available to the designer and the ranges of their properties. Search routines could then be developed to help a designer find design elements to meet specific functional requirements. For software design the design elements would consist of algorithms, precoded if possible with standard input/output interfaces, and data structures implemented as sets of routines for storing, modifying and altering individual items of data.

For the designer, the utility of such a system would be manifested on two levels. In some domains, where most of the designing is done by instantiating and modifying standard designs the system would provide a direct aid to better, faster designs. In domains where standard designs are less useful due to variations in requirements and improvements in technology such a system would allow the designer to study existing designs quickly and

comprehensively so that he can create new designs based on a thorough understanding of the state of the art.

Consider how such a facility could assist in the design of a business application system such as for invoicing. The user would ask the facility what it knew about invoicing systems. The facility would respond with an offer to provide details of generic operations in invoicing systems as well as the details of specific systems that it knows about. The user could look at all or some of this information to decide on the requirements of the information systems that will fit his needs.

If his requirements are not substantially at variance with the properties of one the systems known to the facility then he may decide to compromise a little and adopt an available system. If the requirements are not very different but cannot be compromised the user could examine the structure of the existing system to determine where modifications would need to be made, what form they would take and how extensive they would need to be. If the differences between the requirements and the properties of the existing systems were still larger, the user would have to generate his own system. The general information on invoicing systems and the structure of the systems known to the facility would help him decide on the architecture of the system. Once the function of the modules in the architecture was specified, the information facility would be used again to determine whether it had information about any modules that performed the required functions and if so, what their specific features were. Thus, at this stage, the user would use the information facility for each module as he had used it earlier for the entire system. This process would continue until he was able to specify the entire system in terms of existing or specially developed modules at various levels.

Section 6 describes the operation of such an information facility from another viewpoint.

5.3 Aids to Design Evaluation

While an information facility for design generation, such as the one described above, can be built with essentially similar architecture for any design domain the testing of

proposed designs is more difficult and presents considerably more variety. Essentially, the designs must be represented in a formal manner (see Carroll, Thomas and Malhotra, 1978) and the properties of the design derived mathematically from the design specifications. These properties can then be tested against the functional requirements.

Systems to test proposed designs have been built for domains such as alloy design (Iwata, Ishino and Mishima, 1975) and the design of industrial control systems (Prunet, Floutier and Dumas, 1975) in which designs can be represented mathematically. Less structured domains such as architecture and software design will have to rely on some sort of simulation to test proposed designs. The user will not be able to test the properties of the actual system but he may be able to work with a model that duplicates the important properties of the design. In software design, for example, a facility that would simulate a design from a formal specification and allow a user to try it out would be of considerable value. The user would be able to test whether the design did what it was supposed to do and, more importantly, whether the syntax of the command language and the other human factors of the system were right. Areas such as human factors, where requirements are hard to formalize, are often critical to the success of a product or a system. In many cases a user will be able to tell whether or not a design is acceptable even though he may not be able to articulate the basis on which he makes the decision. A facility that allows proposed designs to be tested along these dimensions should, therefore, be very valuable to the designer.

Conclusions

A study of the design process in a variety of situations has led to a set of proposed aids to for different component design processes. The relationship between the empirical results and the suggested aids are diagrammed in Figure 2.

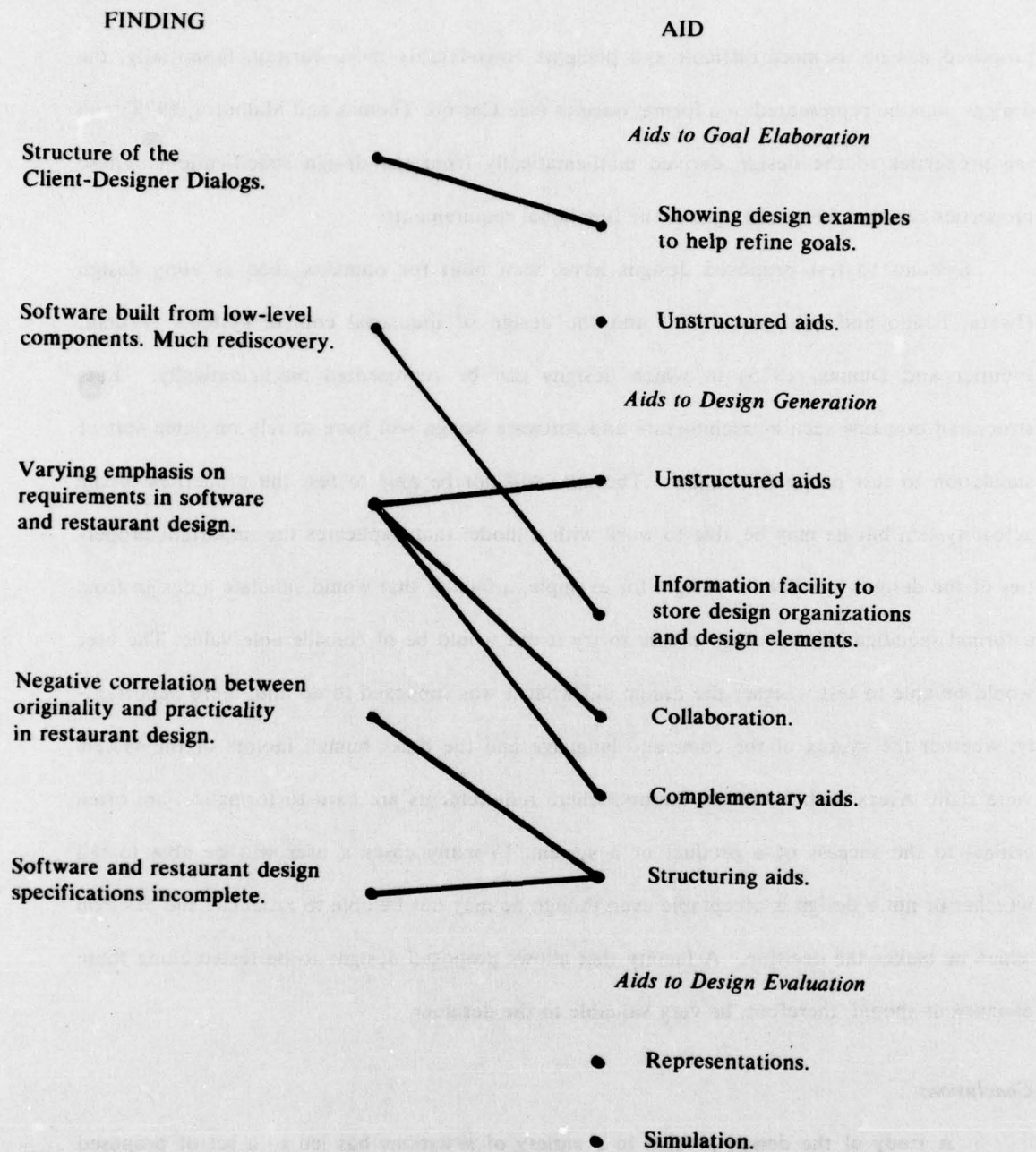


Figure 2. Relationship Between Empirical Findings and Proposed Aids.

6. RECOMMENDATIONS FOR FURTHER ACTION

From an engineering standpoint we recommend that two kinds of systems to aid the design process be investigated further. Unstructured aids seem to have the potential for assisting clients in the goal elaboration process and the designer during design generation. These aids have, however, been insufficiently investigated. Further research should be directed towards perfecting the detailed structures of these aids and demonstrating that specific aids, under controlled circumstances, improve performance on design problems.

The other major direction of work should be towards the implementation of a domain-specific design-support system along the lines of the information facility discussed in section 5.2.6. Such a system would be able to present examples of realized designs to assist in goal elaboration. It would also contain design organizations in the sense of general architectures for common applications and design elements that carry out different kinds of functions to assist in design generation. Each of these would be classified in various ways and could be retrieved on an associational basis from partial requirements stated in a variety of ways.

To set up such a system for a particular branch of design, such as software design, would involve a great deal of initial work in classifying and structuring known designs and design organizations. Once this initial work was done, however, new designs would have to be disclosed in the format prescribed by the system. This would constitute a form of publication. The description of the design would have to be in machine-readable form and would consist of a hierarchical structure of modules. Each module would describe a particular aspect of the system: authors, contents, purpose, comparison with related designs, top-level organization, details of individual modules, performance characteristics, implementation details, system personnel, etc. Modules would contain directories of other modules with related information giving the name of the module and the information contained in it. This structure would enable the user to access the information to whatever depth he desires. Further, it would impose standards on the author and insure that he included all the required information.

Such a system could be very valuable to software designers and if it were properly implemented it could become an indispensable companion to their daily work. Inclusion of information about a piece of software in such a system would then become more important than publication in a journal. This would serve as an incentive for the author to follow the prescribed format and provide all the information required by the system. Further, access to individual designs and design elements could be limited and monitored and royalties charged for the use of information contained therein. Royalties could be charged at a higher rate for access to the machine readable implementation of the modules. Thus, the designer could obtain information as well as the equivalent of off-the-shelf assemblies from such a system.

Through the use of tools such as these, software design cost may be substantially reduced, the software development process may be made more predictable and software reliability may be improved.

REFERENCES

- Alexander, C. *Notes on the Synthesis of Form*. Cambridge: Harvard University Press, 1964.
- Amkreutz, J. H. A. E. Cybernetic model of the design process, *Computer Aided Design*, 1976, 8 (3), 187-192.
- Carroll, J. M., Thomas, J. C. & Malhotra, A. Presentation and Representation in Design Problem Solving. IBM Research Report RC 6975, 1978
- Duncker, K. On Problem-Solving, *Psychological Monographs*, 58 (5) 1945.
- Eastman, C. M. Cognitive processes and ill-defined problems: a case study from design. *Proceedings of First Joint International Conference on Artificial Intelligence*, Washington D. C., 1969.
- Forsyth, F., *The Day of the Jackal*. The Viking Press, N.Y. N.Y., 1971
- Freeman, P. and Newell, A. A model for functional reasoning in design. *Second International Joint Computer Conference on Artificial Intelligence*, London, 1971.
- Iwata, S., Ishino, S. & Mishima, Y., Computer Aided Alloy Designing, *Proc. 2nd. USA-JAPAN Computer Conference*, 1975.
- Jones, J. C., *Design Methods*. N.Y. N.Y.: John Wiley, 1970
- Knuth, D. *Fundamental Algorithms*. Reading, Mass.: Addison-Wesley, 1969a.
- Knuth, D. *Seminumerical Algorithms*. Reading, Mass.: Addison-Wesley, 1969b.
- Loewner, P. G., Federated Programming Capabilities, IBM Research Report, RC 2697, 1969
- Meister, D. *Behavioral Foundations of System Development*. New York: Wiley, 1976.
- Metzger, P. W., *Managing a Programming Project*. Englewood Cliffs, N.J.: Prentice-Hall, 1973
- Miller, L. A. and Becker, C., Programming in Natural English, IBM Research Report 5137, 1974
- Morrison, J. P., Data Responsive, Modular, Interleaved Task Programming System, IBM Technical Disclosure Bulletin CA.8.70.10010, Vol 13 No. 8, pp 2425-6, 1971
- Newell, A. and Simon, H. A., *Human Problem Solving*. Englewood Cliffs, New Jersey: Prentice-Hall, 1972.
- Papanek, V., *Design for the Real World*. N.Y. N.Y.: Pantheon Books, 1971
- Prince, G. M. *The Practice of Creativity*. New York: Harper, 1970.
- Prunet, F., Floutier, D. & Dumas, J. M., Computer Aided Design of Industrial Control Systems, *Proc. 12th. IEEE Design Automation Conference*, 1975.

- Reitman, W. *Cognition and Thought*, N.Y. N.Y.: Wiley, 1965.
- Scott, R. F. & Simmons, D. B., Programmer productivity and the Delphi technique, *Datamation* 1974 20 (5), 71-73.
- Simon, H. A. The structure of ill-structured problems, *Artificial Intelligence*, 1973, 4, 181-201.
- Stein, M. *Stimulating Creativity*. New York: Academic Press, 1974.
- Thomas, J. C., An Analysis of Behavior in the Hobbits-Orcs Problem, *Cognitive Psychology*, 1974, 6, 257-269.
- Thomas, J. C. & Lyon, D. IBM Research Report, in preparation, 1978
- Thomas, J. C., Lyon, D. & Miller, L. A., Aids for Problem-Solving, IBM Research Report RC 6468, 1977.
- Thomas, J. C., Malhotra, A. and Carroll, J. C., An Experimental Investigation of the Design Process, IBM Research Report RC 6702, 1977.
- Walker, B., Gurd, J., and Drawneek, E. *Interactive Computer Graphics*, New York: Crane Russak, 1975.
- Walston, C. E. & Felix, C. P., A method of programming estimation and measurement, *IBM Systems Journal*, 16 (1) 54-73.
- Weizenbaum, J., ELIZA --- A Computer Program For the Study of Natural Language Communication Between Man and Machine, *Comm. ACM*, 9 (1) 36-45, 1966.
- Zloof, M., Query By Example: A data base language, *IBM Systems Journal*, 1977.

OFFICE OF NAVAL RESEARCH, CODE 455
TECHNICAL REPORTS DISTRIBUTION LIST

• Director, Engineering Psychology
 Programs, Code 455
 Office of Naval Research
 800 North Quincy Street
 Arlington, VA 22217 (5 cys)

Defense Documentation Center
 Cameron Station
 Alexandria, VA 22314 (12 cys)

Dr. Robert Young
 Director, Cybernetics Technology Office
 Advanced Research Projects Agency
 1400 Wilson Blvd.
 Arlington, VA 22209

Col. Henry L. Taylor, USAF
 OAD (E&LS) ODDR&E
 Pentagon, Room 3D129
 Washington, D.C. 20301

Office of Naval Research
 International Programs
 Code 102IP
 800 North Quincy Street
 Arlington, VA 22217 (6 cys)

Director, Information Systems
 Program, Code 437
 Office of Naval Research
 800 North Quincy Street
 Arlington, VA 22217

Commanding Officer
 ONR Branch Office
 ATTN: Dr. J. Lester
 495 Summer Street
 Boston, MA 02210

Commanding Officer
 ONR Branch Office
 ATTN: Dr. Charles Davis
 536 South Clark Street
 Chicago, IL 60605

• Commanding Officer
 ONR Branch
 ATTN: Dr. E. Gloye
 1030 East Green Street
 Pasadena, CA 91106

Dr. B. McDonald
 Office of Naval Research
 Scientific Liaison Group
 American Embassy, Room A-407
 APO San Francisco 96503

Director, Naval Research
 Laboratory
 Technical Information Division
 Code 2627
 Washington, D.C. 20375 (6 cys)

Office of the Chief of Naval
 Operations, OP987P10
 Personnel Logistics Plans
 Department of the Navy
 Washington, D.C. 20350

CDR Paul Nelson
 Naval Medical R&D Command
 Code 44
 Naval Medical Center
 Bethesda, MD 20014

Director
 Behavioral Sciences Department
 Naval Medical Research Institute
 Bethesda, MD 20014

Dr. George Moeller
 Human Factors Engineering Branch
 Submarine Medical Research
 Laboratory
 Naval Submarine Base
 Groton, CT 06340

Bureau of Naval Personnel
 Special Assistant for Research
 Liaison
 PERS-OR
 Washington, D.C. 20370

Navy Personnel Research and
 Development Center
 Management Support Department
 Code 210
 San Diego, CA 92152

Naval Training Equipment Center
 ATTN: Technical Library
 Orlando, FL 32813

Human Factors Department
Code N215
Naval Training Equipment Center
Orlando, FL 32813

Dr. Alfred F. Smode
Training Analysis and
Evaluation Group
Naval Training Equipment Center
Code N-OOT
Orlando, FL 32813

Dr. Gary Poock
Operations Research Department
Naval Postgraduate School
Monterey, CA 93940

Dr. A. L. Slafkosky
Scientific Advisor
Commandant of the Marine Corps
Code RD-1
Washington, D.C. 20380

Mr. J. Barber
Headquarters, Department of the
Army, DAPE-PBR
Washington, D.C. 20546

Dr. Joseph Zeidner
Director, Organization and
Systems Research Laboratory
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Edgar M. Johnson
Organization and Systems
Research Laboratory
U.S. Army Research Lab
5001 Eisenhower Avenue
Alexandria, VA 22333

Technical Director
U.S. Army Human Engineering Labs
Aberdeen Proving Ground
Aberdeen, MD 21005

U.S. Air Force Office of Scientific
Research
Life Sciences Directorate, NL
Bolling Air Force Base
Washington, D.C. 20332

Dr. Donald A. Topmiller
Chief, Systems Engineering Branch
Human Engineering Division
USAF AMRL/HES
Wright-Patterson AFB, OH 45433

Lt. Col. Joseph A. Birt
Human Engineering Division
Aerospace Medical Research
Laboratory
Wright Patterson AFB, OH 45433

Air University Library
Maxwell Air Force Base, AL 36112

Dr. Alphonse Chapanis
The Johns Hopkins University
Department of Psychology
Charles & 34th Streets
Baltimore, MD 21218

Dr. Arthur I. Siegel
Applied Psychological
Services, Inc.
404 East Lancaster Street
Wayne, PA 19087

Dr. Gerson Weltman
Perceptronics, Inc
6271 Variel Avenue
Woodland Hills, CA 91364

Dr. J. A. Swets
Bolt, Beranek and Newman, Inc.
50 Moulton Street
Cambridge, MA 02138

Dr. H. Rudy Ramsey
Science Applications, Inc.
40 Denver Technological
Center West
7935 East Prentice Avenue
Englewood, Colorado 80110

Dr. Meredith Crawford
5606 Montgomery Street
Chevy Chase, MD 20015

Professor Douglas E. Hunter
Defense Intelligence School
Washington, D.C. 20374

Dr. James H. Howard
Catholic University
Department of Psychology
Washington, D.C. 20064

Dr. Jesse Orlansky
Institute for Defense Analyses
400 Army-Navy Drive
Arlington, VA 22202

Journal Supplement Abstract
Service
American Psychological
Association
1200 17th Street, N.W.
Washington, D.C. 20036 (3 cys)

Dr. William A. McClelland
Human Resources Research Office
300 N. Washington Street
Alexandria, VA 22314

Dr. Tom Love
General Electric Company
Information Systems Programs
1755 Jefferson Davis Highway
Arlington, VA 22202

Dr. David J. Getty
Bolt, Beranek & Newman
50 Moulton Street
Cambridge, MA 02139

Director, Human Factors Wing
Defence & Civil Institute of
Environmental Medicine
Post Office Box 2000
Downsville, Toronto, Ontario
CANADA

Dr. A. D. Baddeley
Director, Applied Psychology Unit
Medical Research Council
15 Chaucer Road
Cambridge, CB2 2EF
ENGLAND